REMARKS

By this amendment, Claims 1-41 are canceled and Claims 42 – 59 are added. No new matter is added. Therefore, Claims 42 – 59 are pending in the application.

Each issued raised in the Office Action mailed March 27, 2008 is addressed hereinafter.

CLAIM REJECTIONS – 35 U.S.C § 103

Canceled claims 1-41 were rejected under 35 U.S.C. § 103(a) as allegedly unpatentable over Barrrett, Jr. et al. (US 6,473,772) (hereinafter "*Barrett*") in view of Yalamanchi et al. (US 2003/0212670) (hereinafter "*Yalamanchi*") and Ling Liu et al. ("Continual Queries for Internet Scale Event-Drive Information Delivery", IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 4, Jul/Aug 1999, pp. 610-628) (hereinafter "*Liu*"), and Kumar et al. (US 7,149,738) (hereinafter "*Kumar*").

Since Claims 1-41 are canceled by this amendment the rejections of those claims made in the Office Action are not addressed in this paper. Applicant respectfully submits, however, that Claims 42 – 59 added by this amended are patentable over any combination of *Barrett*, *Yalamanchi*, *Liu*, and *Kumar*.

CLAIMS 42 AND 51

Claim 42 features a method for managing event-condition-action rules in a database system, the method comprising the computer-implemented steps of:

storing data in a database that represents a composite event comprised of two or more primitive events, at least one condition related to the composite event, and at least one action related to the composite event;

detecting a first database event as an occurrence of a first one of the primitive events;

determining whether the first database event satisfies the at least one condition related to the composite event;

persistently storing the result of the determining in the database;

detecting a second database event as an occurrence of a second one of the primitive events; and

determining whether the at least one condition is satisfied based on the persistently stored result and the second database event.

The method of Claim 42 is fully supported by Applicant's specification including at least by paragraph [0028] and paragraphs [0081] through [0088] along with FIG. 2. The method of Claim 42 enables more efficient processing of event-condition-action (ECA) rules in a database system than can be provided by middleware or application-tier ECA rules engines. Specifically, because a database system that performs the method of Claim 42 supports composite events and persistent storage of incremental evaluation of conditions with respect to primitive events that make up a composite event, there is no restriction on the size of the rule sets or the number of events that can be processed in the database system. *See Specification*, ¶ [0088]. None of *Barrett*, *Yalamanchi*, *Lui*, or *Kumar* teaches or suggests a database system that performs the method of Claim 42.

While *Barrett* describes "rules" for associating "effect" events with "cause" events and *Yalamanchi* describes "Event-Condition-Action (ECA)" rules neither describes composite events. In fact, in rejecting now canceled Claims 3-6 which recited features related to a "composite event structure" the Office Action did not rely on *Barrett* or *Yalamanchi* to satisfy the "composite event structure" feature of those claims. Instead, the Office Action relied on *Kumar* for the claimed "composite event structure." *See Office Action*, page 8-10. However, *Kumar* does not disclose anything like the method of Claim 42 for performing incremental evaluation of conditions with respect to primitive events that make up a composite event. Specifically, *Kumar* does not teach or suggest at least the following bolded features of Claim 42:

detecting a first database event as an occurrence of a first one of the primitive events;

determining whether the first database event satisfies the at least one condition related to the composite event;

persistently storing the result of the determining in the database;

detecting a second database event as an occurrence of a second one of the primitive events; and

determining whether the at least one condition is satisfied based on the persistently stored result and the second database event.

Kumar describes a policy execution layer 106 responsible for executing policies deployed in the system described in *Kumar*. *See Kumar*, Fig. 1. However, there is nothing in *Kumar* that suggests that the policy execution layer stores, **persistently in a database**, the results of determining whether a first event satisfies a policy condition for use in later evaluation of the same condition with respect to the occurrence of a second event.

Kumar at col. 9:18-28 states with respect evaluating conditions associated with a policy:

> On detecting an event, policy execution engine 106 searches for the required event identifier and the corresponding policy in policy database 108. Thereafter, **condition evaluator 416** determines the condition associated with the event. In case of multiple conditions associated for a particular event, each condition may be evaluated concurrently. If the condition is evaluated to be true, **conditional evaluator 416** notifies action performer 418 for action execution. Finally, on successful event detection and condition evaluation, action performer 418 executes the actions defined in the policy.

Kumar at col. 13:8-27 states:

> Once a relevant event is detected, its identifier (such as event ID) information is sent to **condition evaluator 416**. **Condition evaluator 416** then pulls out all applicable policies (conditions and actions) that are stored in the rule table in policy database 108. It may simply use event ID and get the corresponding condition and action strings from the rule table. **Condition evaluator 416** now checks the condition corresponding to each policy. These conditions may relate to external systems as well as to resources, such as system databases. Typically, the conditions are simple logic operations, such as comparison of two values that characterize an external system or an internal system (such as a database). Thus, **condition evaluator 416** receives inputs from resources within and outside the system. Condition evaluator may also execute system commands (for e.g. using SQL query), in order to evaluate a condition (for e.g. to check the status of

> resource). After condition evaluation, **condition evaluator 416** either rejects the rule (if the condition is not true), or passes the action part of the rule to action performer 418 (if the condition is true).

Nowhere in these portions or elsewhere in *Kumar* is it described that the condition evaluator 416 or another component described in *Kumar* persistently stores the results of evaluating a condition in a database. It appears only that the policy execution layer in *Kumar* stores the results of condition evaluation in volatile memory. Thus, unlike the method of Claim 42, the policy execution layer of *Kumar* is limited by the amount of volatile memory available to hold condition evaluation results. Consequently, it is respectfully submitted that **Kumar** does not teach or suggest "persistently storing the result of the determining in the database" as featured in Claim 42.

Further, since *Kumar* does not persistently stores results of condition evaluation in a database it cannot and does not describe the following feature of Claim 42 related to determining whether a condition related to a composite event is satisfied:

> determining whether the at least one condition is satisfied **based on the persistently stored result** and the second database event.

Consequently, it is respectfully submitted that Claim 42 is patentable over *Kumar*.

In rejecting now canceled Claim 4 the Office Action contends that the feature of Claim 4 reciting "persistently storing results of said determining in said database" is satisfied by the Abstract of *Barrett*. Applicants disagree with the Office Action's contention. The "determining referred to in this feature of Claim 4 involves determining whether an event satisfies a condition related to a composite event structure. The Abstract of *Barrett* states that "[t]he driving of events is mediated by a library process that receives observed events from the monitors, in the form of data structures, store[s] them in a database, and passes the effects to be driven to the

appropriate driver in accordance with the set of rules, also data structures stored in the database, when a cause corresponds to a observed event." Thus *Barrett* describes storing data structures that correspond to observed events and rules that map cause events to effect events.

However, Claim 42 is not merely about storing events and conditions in a database. Claim 42 also features persistently storing **the result** of determining whether an event satisfies a condition related to a **composite event**. *Barrett* does not appear to persistently store the result of determining whether an observed event satisfies a rule for the purpose of incremental evaluation of the rule. Further, *Barrett* does not describe anything like the claimed "composite event." Therefore, *Barrett* does not teach or suggest "persistently storing the result of the determining in the database" as featured in Claim 42.

Based on the foregoing, it is respectfully submitted that Claim 42 is allowable over *Barrett*, *Yalamanchi*, Liu, and *Kumar*.

Claim 51 is a computer-readable volatile or non-volatile storage medium counterpart to Claim 42 and recites features similar to those recited in Claim 42. Claim 51, therefore, is allowable for the same reasons that Claim 42 is allowable.

REMAINING CLAIMS

The pending claims not discussed so far are dependant claims that depend on an independent claim that is discussed above. Because each dependant claim includes the features of claims upon which they depend, the dependant claims are patentable for at least those reasons the claims upon which the dependant claims depend are patentable. Removal of the rejections with respect to the dependant claims and allowance of the dependant claims is respectfully requested. In addition, the dependent claims introduce additional features that independently

render them patentable. Due to the fundamental differences already identified, a separate discussion of those features is not included at this time.

CONCLUSIONS

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

The Examiner is respectfully requested to contact the undersigned by telephone if it is believed that such contact would further the examination of the present application.

Please charge any shortages or credit any overages to Deposit Account No. 50-1302.

Respectfully submitted,

Hickman Palermo Truong & Becker LLP

Date: July 28, 2008

/AdamCStone#60531/
Adam Christopher Stone
Reg. No. 60,531

2055 Gateway Place, Suite 550
San Jose, California 95110-1083
Telephone No.: (408) 414-1231
Facsimile No.: (408) 414-1076